H 5

```
FFFFFFFFFFFFFFF         111            111      XXX         XXX
FFFFFFFFFFFFFFF         111            111      XXX         XXX
FFFFFFFFFFFFFFF         111            111      XXX         XXX
FFF                  111111         111111      XXX         XXX
FFF                  111111         111111      XXX         XXX
FFF                  111111         111111      XXX         XXX
FFF                     111            111         XXX   XXX
FFF                     111            111         XXX   XXX
FFF                     111            111         XXX   XXX
FFFFFFFF.FFF            111            111            XXX
FFFFFFFFFFFF            111            111            XXX
FFFFFFFFFFFF            111            111            XXX
FFF                     111            111         XXX   XXX
FFF                     111            111         XXX   XXX
FFF                     111            111         XXX   XXX
FFF                     111            111      XXX         XXX
FFF                     111            111      XXX         XXX
FFF                     111            111      XXX         XXX
FFF                  111111111      111111111   XXX         XXX
FFF                  111111111      111111111   XXX         XXX
FFF                  111111111      111111111   XXX         XXX
```

```
_$25

Symb
----
IOC1
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_K
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK
```

```
**FILE**ID**FILUTL                                    G  9
```

```
FFFFFFFFFF    IIIIII    LL          UU      UU  TTTTTTTTTT  LL
FFFFFFFFFF    IIIIII    LL          UU      UU  TTTTTTTTTT  LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FFFFFFFF        II      LL          UU      UU      TT      LL
FFFFFFFF        II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL              ....
FF            IIIIII    LLLLLLLLLL  UUUUUUUUUU      TT      LLLLLLLLLL      ....
FF            IIIIII    LLLLLLLLLL  UUUUUUUUUU      TT      LLLLLLLLLL      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

FIL
VO4

```
0 MODULE FILUTL (
0                       LANGUAGE (BLISS32),
0                       IDENT = 'V04-000'
0                       ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  F11ACP Structure Level 2
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains routines used to access random files by the
1 !       ACP itself.
1 !
1 ! ENVIRONMENT:
1 !
1 !       STARLET operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  22-May-1978  19:13
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-015 CDS0010         Christian D. Saether    14-Aug-1984
1 !               Modify handling of extension fcbs.
1 !
1 !       V03-014 CDS0009         Christian D. Saether     6-Aug-1984
1 !               Correctly deal with serializing on a lock we already had.
1 !               Add handler for the open_file routine to correctly
```

```
1 !                     clean up after errors in the open_file routine.
1 !
1 !             V03-013 LMP0275         L. Mark Pilant,         25-Jul-1984  15:50
1 !                     Don't try to delete an uninitialized ACL.
1 !
1 !             V03-012 CDS0008         Christian D. Saether    19-Apr-1984
1 !                     Use REFCNT instead of ACNT.
1 !                     Modify access arbitration.
1 !
1 !             V03-011 ACG0415         Andrew C. Goldstein,    5-Apr-1984  21:33
1 !                     Interface change to ACL_DELETEACL
1 !
1 !             V03-010 ACG0408         Andrew C. Goldstein,    20-Mar-1984  17:47
1 !                     Make APPLY_RVN and DEFAULT_RVN macros
1 !
1 !             V03-009 CDS0007         Christian D. Saether    23-Feb-1984
1 !                     Eliminate use of FLUSH_LOCK_BASIS.
1 !                     Replace with TOSS_CACHE_DATA.
1 !
1 !             V03-008 CDS0006         Christian D. Saether    18-Jan-1984
1 !                     Modify interface to APPLY_RVN.
1 !
1 !             V03-007 CDS0005         Christian D. Saether    30-Dec-1983
1 !                     Use L_NORM linkage and BIND_COMMON macro.
1 !
1 !             V03-006 CDS0004         Christian D. Saether     7-Dec-1983
1 !                     Remove call to REMOVE_FCB and do the REMQUE inline.
1 !
1 !             V03-005 CDS0003         Christian D. Saether    14-Sep-1983
1 !                     Modify SERIAL_FILE interface.  Use RELEASE_SERIAL_LOCK
1 !                     routine to dequeue serialization lock.
1 !
1 !             V03-004 CDS0002         Christian D. Saether    19-Jun-1983
1 !                     Until further work is done with buffer caching,
1 !                     flush all buffers from the cache when closing internal file.
1 !                     This fixes a bug where getting location information for
1 !                     VBN placement leaves a header in the cache and the file
1 !                     serialization lock is released.
1 !
1 !             V03-003 CDS0001         Christian D. Saether     5-May-1983
1 !                     Add xqp synchronization of file processing (SERIAL_FILE)
1 !                     and xqp access arbitration (ACCESS_LOCK) calls.
1 !
1 !             V03-02  LMP0059         L. Mark Pilant,          7-Jan-1983  12:05
1 !                     Always create and link in an FCB when accessing a file.  This
1 !                     eliminates a lot of special case handling.
1 !
1 !             V03-001 LMP0037         L. Mark Pilant,         28-Jun-1982  15:10
1 !                     Remove the addressing mode module switch.
1 !
1 !             V02-006 ACG0259         Andrew C. Goldstein,    27-Jan-1982  20:38
1 !                     Change to longword external addressing
1 !
1 !             V02-004 LMP0003         L. Mark Pilant,          8-Dec-1981  11:31
1 !                     Make sure the primary window was actually created.  It may
1 !                     not have been due to the byte limit quota being exceeded.
1 !
```

```
1 !          B0104    ACG0112         Andrew C. Goldstein,    15-Jan-1980  22:55
1 !                   Limit data read to file's EOF
1 !
1 !          B0103    ACG0092         Andrew C. Goldstein,    6-Dec-1979  19:23
1 !                   Set proper RVN on file being opened
1 !
1 !          B0102    ACG0008         Andrew C. Goldstein,  18-Dec-1978  22:57
1 !          Add map only access for placement use, support multi-header files
1 !
1 !          B0101    ACG0003         Andrew C. Goldstein,  10-Nov-1978  19:01
1 !          Add multi-volume support, restrict to single header files
1 !
1 !          B0100    ACG00001        Andrew C. Goldstein,  10-Oct-1978  20:00
1 !          Previous revision history moved to [F11B.SRC]F11B.REV
1 !**
1
1
1 LIBRARY 'SYS$LIBRARY:LIB.L32';
1 REQUIRE 'SRC$:FCPDEF.B32';
1
1
1 FORWARD ROUTINE
1          OPEN_FILE          : L_NORM,        ! open a file
1          OPEN_FILE_HANDLER  : L_NORM,        ! error handling for open_file
1          READ_DATA          : L_NORM,        ! read data from file
1          CLOSE_FILE         : L_NORM NOVALUE; ! close a file
```

```
1 GLOBAL ROUTINE OPEN_FILE (FID, WRITE) : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine opens the file of the given file ID. It constructs an
1 !       FCB and window and returns the address of the latter.
1 !
1 !
1 ! CALLING SEQUENCE:
1 !       OPEN_FILE (ARG1, ARG2)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: address of file ID of file to open
1 !       ARG2: = 0 to open read only
1 !               1 to open read/write
1 !               2 to bypass interlocks (just map the file)
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       PRIMARY_FCB: address of FCB created or found
1 !       CURRENT_WINDOW: address of window created
1 !
1 ! ROUTINE VALUE:
1 !       address of window created
1 !
1 ! SIDE EFFECTS:
1 !       FCB and window created
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       FID             : REF BBLOCK;   ! file ID arg
2
2 LOCAL
2       FCB_CREATED,                    ! flag indicating FCB creation
2       FCB             : REF BBLOCK,   ! file control block address
2       WINDOW          : REF BBLOCK,   ! window address
2       HEADER          : REF BBLOCK;   ! file header address
2
2 BIND_COMMON;
2
2 EXTERNAL ROUTINE
2       REBLD_PRIM_FCB  : L_NORM NOVALUE, ! rebuild primary fcb from header
2       BUILD_EXT_FCBS  : L_NORM NOVALUE, ! build extension fcbs
2       ARBITRATE_ACCESS : L_JSB_2ARGS, ! arbitrate file access
2       CONV_ACCLOCK    : L_NORM,       ! convert file access lock
2       SERIAL_FILE     : L_NORM,       ! file processing interlock
2       SWITCH_VOLUME   : L_NORM,       ! switch to correct volume
```

```
2               SEARCH_FCB      : L_NORM,       ! search for FCB of file
2               READ_HEADER     : L_NORM,       ! read file header
2               CREATE_FCB      : L_NORM,       ! create a file control block
2               CREATE_WINDOW   : L_NORM;       ! create a file window
2
2 ENABLE OPEN_FILE_HANDLER;
2
2 ! The current uses of this routine (as of 3b) are
2 ! 1) BADSCN calls it to get r/w access to the badlog file
2 ! 2) GET_LOC calls it with bypass to get mapping info for related file placement
2 ! 3) CREATE calls it with bypass to get previous version attributes for
2 ! propagation
2 !
2 ! There is a small possibility of deadlock on the placement use because of
2 ! the file serialization lock.  If two processes simultaneously do placed
2 ! allocation on two separate files, and each specifies the other as the
2 ! file to be placed near, one could deadlock.
2 !
2
2 ! Initialize impure cells that drive the cleanup in the handler.
2 !
2
2 STSFLGS [STS_HAD_LOCK] = 0;
2 STSFLGS [STS_KEEP_LOCK] = 0;
2 PRIMARY_FCB = 0;
2 PRIM_LCKINDX = 0;
2
2 ! Switch context to the volume of the specified RVN.
2 !
2
2 APPLY_RVN (FID[FID$W_RVN], .CURRENT_RVN);
2 SWITCH_VOLUME (.FID[FID$W_RVN]);
2
2 ! Interlock processing on this file.
2 ! There is an assumption made in the way that this lock is handled
2 ! that no other serial_file calls will be made before a close_file
2 ! is done on this file.  That is because the sts_had_lock flag will
2 ! be set by serial_file and we are going to use that flag to determine
2 ! whether to release this lock in close_file.
2 !
2
2 PRIM_LCKINDX = SERIAL_FILE (.FID);
2
2 IF .STSFLGS [STS_HAD_LOCK]
2 THEN
2     STSFLGS [STS_KEEP_LOCK] = 1;
2
2 ! Search the FCB list for the given file ID. If found, arbitrate access
2 ! interlocks. Note that if we create an FCB, we do not bother with access
2 ! counts, etc., since it will disappear at the end of this call.
2 !
2
2 FCB = SEARCH_FCB (.FID);
2
2 HEADER = READ_HEADER (.FID, .FCB);
2 FCB_CREATED = 0;
2 IF .FCB EQL 0
```

```
2 THEN
3     BEGIN
3     FCB_CREATED = 1;
3     FCB = KERNEL_CALL (CREATE_FCB, .HEADER);
2     END;
2
2 PRIMARY_FCB = .FCB;
2
2 IF .WRITE NEQ 2
2 THEN
3     BEGIN
3     LOCAL
3         CURR_LKMODE;
3
3     CURR_LKMODE = .FCB [FCB$B_ACCLKMODE];
3
3     IF NOT ARBITRATE_ACCESS (IF .WRITE THEN FIB$M_WRITE ELSE 0, .FCB)
3     THEN ERR_EXIT (SS$_ACCONFLICT);
3
3     CONV_ACCLOCK (.CURR_LKMODE, .FCB);
2     END;
2
2 ! By setting this cleanup flag, further error recovery is done in
2 ! the error_cleanup routine, not by the open_file_handler.
2 !
2
2 CLEANUP_FLAGS[CLF_CLOSEFILE] = 1;
2
2 CURRENT_WINDOW = WINDOW = CREATE_WINDOW (0, 0, .HEADER, 0, .FCB);
2
2 IF .CURRENT_WINDOW EQL 0 THEN ERR_EXIT (SS$_EXBYTLM);
2
2 ! If the file is multi-header, read the extension headers and create
2 ! extension FCB's as necessary. Finally read back the primary header.
2 !
2
2 IF .FCB_CREATED
2 THEN
2     BUILD_EXT_FCBS (.HEADER)
2 ELSE
2     IF .FCB [FCB$V_STALE]
2     THEN
3         BEGIN
3
3         REBLD_PRIM_FCB (.FCB, .HEADER);
3
3         BUILD_EXT_FCBS (.HEADER);
3
2         END;
2
2 RETURN .WINDOW;
2
1 END;                                      ! end of routine OPEN_FILE

                                                    .TITLE  FILUTL
                                                    .IDENT  \V04-000\
                                                     .EXTRN  REBLD_PRIM_FCB, BUILD_EXT_FCBS
                                                     .EXTRN  ARBITRATE_ACCESS
                                                     .EXTRN  CONV_ACCLOCK, SERIAL_FILE
                                                     .EXTRN  SWITCH_VOLUME, SEARCH_FCB
                                                     .EXTRN  READ_HEADER, CREATE_FCB
                                                     .EXTRN  CREATE_WINDOW

                                                     .PSECT  $CODE$,NOWRT,2

                                 003C 00000          .ENTRY  OPEN_FILE, Save R2,R3,R4,R5             ; 1131
           6D    00E2    CF  DE 00002          MOVAL   12$, (FP)                              ; 1177
     A6    AA            06  8A 00007          BICB2   #6, -90(BASE)                          ; 1211
                   08    AA  D4 0000B          CLRL    8(BASE)                                ; 1212
                   18    AA  D4 0000E          CLRL    24(BASE)                               ; 1213
           50      04    AC  D0 00011          MOVL    FID, R0                                ; 1218
                   04    A0  95 00015          TSTB    4(R0)
                         05  12 00018          BNEQ    1$
     04    A0      A0    AA  90 0001A          MOVB    -96(BASE), 4(R0)
           50      04    AC  D0 0001F 1$:      MOVL    FID, R0
           01      04    A0  91 00023          CMPB    4(R0), #1
                         08  12 00027          BNEQ    2$
                   A0    AA  D5 00029          TSTL    -96(BASE)
                         03  12 0002C          BNEQ    2$
                   04    A0  94 0002E          CLRB    4(R0)
           50      04    AC  D0 00031 2$:      MOVL    FID, R0                                ; 1219
           7E      04    A0  3C 00035          MOVZWL  4(R0), -(SP)
  0000G    CF            01  FB 00039          CALLS   #1, SWITCH_VOLUME
                   04    AC  DD 0003E          PUSHL   FID                                    ; 1229
  0000G    CF            01  FB 00041          CALLS   #1, SERIAL_FILE
     18    AA            50  D0 00046          MOVL    R0, 24(BASE)
04   A6    AA            01  E1 0004A          BBC     #1, -90(BASE), 3$                      ; 1231
     A6    AA            04  88 0004F          BISB2   #4, -90(BASE)                          ; 1233
                   04    AC  DD 00053 3$:      PUSHL   FID                                    ; 1240
  0000G    CF            01  FB 00056          CALLS   #1, SEARCH_FCB
           52            50  D0 0005B          MOVL    R0, FCB
                         52  DD 0005E          PUSHL   FCB                                    ; 1242
                   04    AC  DD 00060          PUSHL   FID
  0000G    CF            02  FB 00063          CALLS   #2, READ_HEADER
           55            50  D0 00068          MOVL    R0, HEADER
                         54  D4 0006B          CLRL    FCB_CREATED                            ; 1243
                         52  D5 0006D          TSTL    FCB                                    ; 1244
                         0D  12 0006F          BNEQ    4$
           54            01  D0 00071          MOVL    #1, FCB_CREATED                        ; 1247
                         55  DD 00074          PUSHL   HEADER                                 ; 1248
  0000G    CF            01  FB 00076          CALLS   #1, CREATE_FCB
           52            50  D0 0007B          MOVL    R0, FCB
     08    AA            52  D0 0007E 4$:      MOVL    FCB, 8(BASE)                           ; 1251
           02      08    AC  D1 00082          CMPL    WRITE, #2                              ; 1253
                         28  13 00086          BEQL    8$
           53      0B    A2  9A 00088          MOVZBL  11(FCB), CURR_LKMODE                   ; 1259
           07      08    AC  E9 0008C          BLBC    WRITE, 5$                              ; 1261
           50    0100    8F  3C 00090          MOVZWL  #256, R0
                         02  11 00095          BRB     6$
                         50  D4 00097 5$:      CLRL    R0
           51            52  D0 00099 6$:      MOVL    FCB, R1
                      0000G  30 0009C          BSBW    ARBITRATE_ACCESS
```

```
          05                 50  E8 0009F         BLBS    R0, 7$                                   ;
                    0800     8F  BF 000A2         CHMU    #2048                                    ; 1262
                                 04 000A6         RET                                              ;
                             52  DD 000A7  7$:    PUSHL   FCB                                      ; 1264
                             53  DD 000A9         PUSHL   CURR_LKMODE                              ;
     0000G   CF              02  FB 000AB         CALLS   #2, CONV_ACCLOCK                         ;
        03   AA              01  88 000B0  8$:    BISB2   #1, 3(BASE)                              ; 1271
                             52  DD 000B4         PUSHL   FCB                                      ; 1273
                             7E  D4 000B6         CLRL    -(SP)                                    ;
                             55  DD 000B8         PUSHL   HEADER                                   ;
                             7E  7C 000BA         CLRQ    -(SP)                                    ;
     0000G   CF              05  FB 000BC         CALLS   #5, CREATE_WINDOW                        ;
             53              50  D0 000C1         MOVL    R0, WINDOW                               ;
        0C   AA              53  D0 000C4         MOVL    WINDOW, 12(BASE)                         ;
                             05  12 000C8         BNEQ    9$                                       ; 1275
                    2A14     8F  BF 000CA         CHMU    #10772                                   ;
                                 04 000CE         RET                                              ;
             0B              54  E8 000CF  9$:    BLBS    FCB_CREATED, 10$                         ; 1281
             0E       23     A2  E9 000D2         BLBC    35(FCB), 11$                             ; 1285
                             24  BB 000D6         PUSHR   #^M<R2,R5>                               ; 1289
     0000G   CF              02  FB 000D8         CALLS   #2, REBLD_PRIM_FCB                       ;
                             55  DD 000DD  10$:   PUSHL   HEADER                                   ; 1291
     0000G   CF              01  FB 000DF         CALLS   #1, BUILD_EXT_FCBS                       ;
             50              53  D0 000E4  11$:   MOVL    WINDOW, R0                               ; 1295
                                 04 000E7         RET                                              ; 1297
                              0000 000E8  12$:    .WORD   Save nothing                             ; 1177
                             7E  D4 000EA         CLRL    -(SP)                                    ;
                             5E  DD 000EC         PUSHL   SP                                       ;
             7E       04     AC  7D 000EE         MOVQ    4(AP), -(SP)                             ;
     0000V   CF              03  FB 000F2         CALLS   #3, OPEN_FILE_HANDLER                    ;
                                 04 000F7         RET                                              ;

; Routine Size:  248 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 ROUTINE OPEN_FILE_HANDLER (SIGNAL, MECHANISM) : L_NORM =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Clean up from aborted open file.  Specifically, get rid of
1 !       the fcb and serialization lock if we did not previously
1 !       hold the serialization lock.
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       SIGNAL  : REF BBLOCK;
2
2 BIND_COMMON;
2
2 EXTERNAL ROUTINE
2       NUKE_HEAD_FCB   : L_NORM NOVALUE, ! cleanup and deallocate prim fcb
2       RELEASE_SERIAL_LOCK : L_NORM NOVALUE,
2       SET_DIRINDX     : L_JSB_1ARG;
2
2 IF .SIGNAL [CHF$L_SIG_NAME] NEQ SS$_CMODUSER
2     OR .CLEANUP_FLAGS [CLF_CLOSEFILE]
2     OR .PRIM_LCKINDX EQL 0
2     OR .STSFLGS [STS_KEEP_LOCK]
2 THEN
2     RETURN SS$_RESIGNAL;
2
2 IF .PRIMARY_FCB NEQ 0
2 THEN
2     IF .PRIMARY_FCB [FCB$W_REFCNT] EQL 0
2     THEN
2         IF NOT SET_DIRINDX (.PRIMARY_FCB)
2         THEN
2             NUKE_HEAD_FCB (.PRIMARY_FCB);
2
2 PRIMARY_FCB = 0;
2
2 IF .PRIM_LCKINDX NEQ 0
2 THEN
2     RELEASE_SERIAL_LOCK (.PRIM_LCKINDX);
2
2 PRIM_LCKINDX = 0;
2
2 SS$_RESIGNAL
2
1 END;            ! of routine OPEN_FILE_HANDLER


                                                   .EXTRN  NUKE_HEAD_FCB, RELEASE_SERIAL_LOCK
                                                   .EXTRN  SET_DIRINDX

                                   000C 00000 OPEN_FILE_HANDLER:
```

```
                                                                .WORD   Save R2,R3                          ; 1298
                          50       04  AC  D0 00002             MOVL    SIGNAL, R0                          ; 1322
             00000424     8F       04  A0  D1 00006             CMPL    4(R0), #1060                        ;
                                       3A  12 0000E             BNEQ    3$                                  ;
                          36       03  AA  E8 00010             BLBS    3(BASE), 3$                         ; 1323
                                   18  AA  D5 00014             TSTL    24(BASE)                            ; 1324
                                       31  13 00017             BEQL    3$                                  ;
        2C         A6     AA           02  E0 00019             BBS     #2, -90(BASE), 3$                   ; 1325
                          50       08  AA  D0 0001E             MOVL    8(BASE), R0                         ; 1329
                                       13  13 00022             BEQL    1$                                  ;
                                   18  A0  B5 00024             TSTW    24(R0)                              ; 1331
                                       0E  12 00027             BNEQ    1$                                  ;
                                     0000G 30 00029             BSBW    SET_DIRINDX                         ; 1333
                          08           50  E8 0002C             BLBS    R0, 1$                              ;
                                   08  AA  DD 0002F             PUSHL   8(BASE)                             ; 1335
                 0000G    CF           01  FB 00032             CALLS   #1, NUKE_HEAD_FCB                   ;
                                   08  AA  D4 00037 1$:         CLRL    8(BASE)                             ; 1337
                                   18  AA  D5 0003A             TSTL    24(BASE)                            ; 1339
                                       08  13 0003D             BEQL    2$                                  ;
                                   18  AA  DD 0003F             PUSHL   24(BASE)                            ; 1341
                 0000G    CF           01  FB 00042             CALLS   #1, RELEASE_SERIAL_LOCK             ;
                                   18  AA  D4 00047 2$:         CLRL    24(BASE)                            ; 1343
                          50     0918  8F  3C 0004A 3$:         MOVZWL  #2328, R0                           ; 1347
                                       04  04 0004F             RET                                         ;

; Routine Size:  80 bytes,    Routine Base:  $CODE$ + 00F8
```

```
GLOBAL ROUTINE READ_DATA (WINDOW, VBN, COUNT) : L_NORM =

!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine reads the specified data block(s) from the file indicated
!       by the given window address. Note that the actual number of blocks
!       read may be less than the number desired due to mapping fragmentation
!       or cache limitations.
!
!
! CALLING SEQUENCE:
!       READ_DATA (ARG1, ARG2, ARG3)
!
! INPUT PARAMETERS:
!       ARG1: window address
!       ARG2: starting VBN to read
!       ARG3: count of blocks to read
!
! IMPLICIT INPUTS:
!       NONE
!
! OUTPUT PARAMETERS:
!       NONE
!
! IMPLICIT OUTPUTS:
!       NONE
!
! ROUTINE VALUE:
!       address of buffer read
!
! SIDE EFFECTS:
!       block read, window may be turned
!
!--

BEGIN

MAP
        WINDOW          : REF BBLOCK;   ! window argument

LOCAL
        FCB             : REF BBLOCK,   ! address of file's FCB
        LBN,                            ! LBN of starting virtual block
        UNMAPPED,                       ! number of desired blocks not mapped
        BUFFER          : REF BBLOCK;   ! address of block read

BASE_REGISTER;

EXTERNAL ROUTINE
        MAP_VBN         : L_NORM,       ! map virtual to logical
        READ_BLOCK      : L_NORM;       ! read a disk block


! Map the VBN to LBN using the supplied window. If the map fails, return a
! zero buffer address.
```

```
;  418          1405  2 !
;  419          1406  2
;  420          1407  2 FCB = .WINDOW[WCB$L_FCB];
;  421          1408  2 IF .VBN GTRU .FCB[FCB$L_EFBLK]
;  422          1409  2 THEN RETURN 0;
;  423          1410  2
;  424          1411  2 LBN = MAP_VBN (.VBN, .WINDOW, .COUNT, UNMAPPED);
;  425          1412  2 IF .LBN EQL -1 THEN RETURN 0;
;  426          1413  2
;  427          1414  2 BUFFER = READ_BLOCK (.LBN, .COUNT - .UNMAPPED, DATA_TYPE);
;  428          1415  2 RETURN .BUFFER;
;  429          1416  2
;  430          1417  1 END;                                    ! end of routine READ_DATA


                                                            .EXTRN  MAP_VBN, READ_BLOCK

                                        0000 00000          .ENTRY  READ_DATA, Save nothing                 ; 1348
                       5E            04   C2 00002          SUBL2   #4, SP                                  ;
                       51       04   AC   D0 00005          MOVL    WINDOW, R1                              ; 1407
                       50       18   A1   D0 00009          MOVL    24(R1), FCB                             ;
                 3C    A0       08   AC   D1 0000D          CMPL    VBN, 60(FCB)                            ; 1408
                                     28   1A 00012          BGTRU   1$                                      ;
                                     5E   DD 00014          PUSHL   SP                                      ; 1411
                                0C   AC   DD 00016          PUSHL   COUNT                                   ;
                                     51   DD 00019          PUSHL   R1                                      ;
                                08   AC   DD 0001B          PUSHL   VBN                                     ;
                  0000G  CF          04   FB 0001E          CALLS   #4, MAP_VBN                             ;
               FFFFFFFF  8F          50   D1 00023          CMPL    LBN, #-1                                ; 1412
                                     10   13 0002A          BEQL    1$                                      ;
                                     04   DD 0002C          PUSHL   #4                                      ; 1414
            7E      0C   AC     04   AE   C3 0002E          SUBL3   UNMAPPED, COUNT, -(SP)                  ;
                                     50   DD 00034          PUSHL   LBN                                     ;
                  0000G  CF          03   FB 00036          CALLS   #3, READ_BLOCK                          ;
                                          04 0003B          RET                                             ; 1415
                                     50   D4 0003C  1$:     CLRL    R0                                      ; 1417
                                          04 0003E          RET                                             ;

; Routine Size:  63 bytes,    Routine Base:  $CODE$ + 0148
```

```
GLOBAL ROUTINE CLOSE_FILE (WINDOW) : L_NORM NOVALUE =

!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine closes the file indicated by the supplied window
!       by releasing the window and FCB.
!
!
! CALLING SEQUENCE:
!       CLOSE_FILE (ARG1)
!
! INPUT PARAMETERS:
!       ARG1: address of window
!
! IMPLICIT INPUTS:
!       NONE
!
! OUTPUT PARAMETERS:
!       NONE
!
! IMPLICIT OUTPUTS:
!       PRIMARY_FCB: 0
!       CURRENT_WINDOW: 0
!
! ROUTINE VALUE:
!       NONE
!
! SIDE EFFECTS:
!       FCB and window deallocated
!
!--

BEGIN

MAP
        WINDOW          : REF BBLOCK;   ! window argument

LOCAL
        FCB             : REF BBLOCK,   ! FCB of file
        WINDOW_SEGMENT  : REF BBLOCK,   ! Address of current window segment
        NEXT_SEGMENT    : REF BBLOCK;   ! Address of next window segment

BIND_COMMON;

EXTERNAL ROUTINE
        TOSS_CACHE_DATA : L_NORM NOVALUE,
        RELEASE_SERIAL_LOCK : L_NORM NOVALUE,
        DEALLOCATE      : L_NORM,       ! deallocate back to pool
        DEL_EXTFCB      : L_NORM,       ! delete extension FCB's
        SET_DIRINDX     : L_JSB_1ARG,   ! test and set for directory fcb
        NUKE_HEAD_FCB   : L_NORM NOVALUE; ! cleanup a primary fcb


! Find the FCB. Deallocate the window, and the FCB if it is not otherwise
! accessed. Also flush data blocks of the file from the buffer pool.
```

```
;  489          1475  2 !
;  490          1476  2
;  491          1477  2 FCB = .WINDOW[WCB$L_FCB];
;  492          1478  2 TOSS_CACHE_DATA (.PRIM_LCKINDX);
;  493          1479  2
;  494          1480  2 PRIMARY_FCB = 0;
;  495          1481  2 CURRENT_WINDOW = 0;
;  496          1482  2 CLEANUP_FLAGS[CLF_CLOSEFILE] = 0;
;  497          1483  2
;  498          1484  2 WINDOW_SEGMENT = .WINDOW;
;  499          1485  2 DO
;  500          1486  3     BEGIN
;  501          1487  3     NEXT_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
;  502          1488  3     KERNEL_CALL (DEALLOCATE, .WINDOW_SEGMENT);
;  503          1489  3     WINDOW_SEGMENT = .NEXT_SEGMENT;
;  504          1490  3     END
;  505          1491  2 UNTIL .WINDOW_SEGMENT EQL 0;
;  506          1492  2
;  507          1493  2 ! If we already held the serialization lock on this file, we must
;  508          1494  2 ! have it in primary context.  In that case, we will also have
;  509          1495  2 ! remembered the fcb address in primary context, so let's deal with
;  510          1496  2 ! it there.
;  511          1497  2 !
;  512          1498  2
;  513          1499  2 IF .STSFLGS [STS_KEEP_LOCK]
;  514          1500  2 THEN
;  515          1501  3     BEGIN
;  516          1502  3     PRIM_LCKINDX = 0;
;  517          1503  3     RETURN;
;  518          1504  3     END;
;  519          1505  2
;  520          1506  2 IF .FCB[FCB$W_REFCNT] EQL 0
;  521          1507  2 THEN
;  522          1508  2     IF NOT SET_DIRINDX (.FCB)
;  523          1509  2     THEN
;  524          1510  3         BEGIN
;  525          1511  3         DEL_EXTFCB (.FCB);
;  526          1512  3         NUKE_HEAD_FCB (.FCB);
;  527          1513  3         END;
;  528          1514  2
;  529          1515  2 RELEASE_SERIAL_LOCK (.PRIM_LCKINDX);
;  530          1516  2 PRIM_LCKINDX = 0;
;  531          1517  2
;  532          1518  1 END;                                          ! end of routine CLOSE_FILE


                                                                    .EXTRN   TOSS_CACHE_DATA
                                                                    .EXTRN   DEALLOCATE, DEL_EXTFCB

                                      001C 00000                    .ENTRY   CLOSE_FILE, Save R2,R3,R4                     ; 1418
                50        04    AC    D0 00002                      MOVL     WINDOW, R0                                    ; 1477
                53        18    A0    D0 00006                      MOVL     24(R0), FCB                                   ;
                          18    AA    DD 0000A                      PUSHL    24(BASE)                                      ; 1478
          0000G CF              01    FB 0000D                      CALLS    #1, TOSS_CACHE_DATA                           ;
                          08    AA    7C 00012                      CLRQ     8(BASE)                                       ; 1480
             03 AA              01    8A 00015                      BICB2    #1, 3(BASE)                                   ; 1482
```

```
                              52       04   AC  D0 00019        MOVL    WINDOW, WINDOW_SEGMENT                              ; 1484
                              54       20   A2  D0 0001D 1$:    MOVL    32(WINDOW_SEGMENT), NEXT_SEGMENT                    ; 1487
                                            52  DD 00021        PUSHL   WINDOW_SEGMENT                                      ; 1488
                         0000G CF           01  FB 00023        CALLS   #1, DEALLOCATE                                      ;
                              52            54  D0 00028        MOVL    NEXT_SEGMENT, WINDOW_SEGMENT                        ; 1489
                                            F0  12 0002B        BNEQ    1$                                                  ; 1491
                 24     A6    AA            02  E0 0002D        BBS     #2, -90(BASE), 3$                                   ; 1499
                                       18   A3  B5 00032        TSTW    24(FCB)                                             ; 1506
                                            17  12 00035        BNEQ    2$                                                  ;
                              50            53  D0 00037        MOVL    FCB, R0                                             ; 1508
                                         0000G  30 0003A        BSBW    SET_DIRINDX                                         ;
                              0E            50  E8 0003D        BLBS    R0, 2$                                              ;
                                            53  DD 00040        PUSHL   FCB                                                 ; 1511
                         0000G CF           01  FB 00042        CALLS   #1, DEL_EXTFCB                                      ;
                                            53  DD 00047        PUSHL   FCB                                                 ; 1512
                         0000G CF           01  FB 00049        CALLS   #1, NUKE_HEAD_FCB                                   ;
                                       18   AA  DD 0004E 2$:    PUSHL   24(BASE)                                            ; 1515
                         0000G CF           01  FB 00051        CALLS   #1, RELEASE_SERIAL_LOCK                             ;
                                       18   AA  D4 00056 3$:    CLRL    24(BASE)                                            ; 1516
                                                04 00059        RET                                                         ; 1518

; Routine Size:  90 bytes,    Routine Base:  $CODE$ + 0187


;  533          1519  1
;  534          1520  1 END
;  535          1521  0 ELUDOM




;                                 PSECT SUMMARY
;
;       Name                        Bytes                        Attributes
;
;  $CODE$                               481  NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)




;                              Library Statistics
;
;                                              -------- Symbols --------       Pages       Processing
;       File                                   Total    Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1               18619        30         0       1000          00:01.9




;                                  COMMAND QUALIFIERS
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:FILUTL/OBJ=OBJ$:FILUTL MSRC$:FILUTL/UPDATE=(ENH$:FILUTL)

; Size:          481 code + 0 data bytes
; Run Time:         00:35.2
; Elapsed Time:     01:31.0
; Lines/CPU Min:     2589
; Lexemes/CPU-Min: 60078
; Memory Used:  229 pages
; Compilation Complete
```

0170 AH-BT13A-SE DIGITAL EQUIPMENT CORPORATION
VAX/VMS V4.0 CONFIDENTIAL AND PROPRIETARY

EXTIDX
LIS

GTLCAT
LIS

GETLOC
LIS

EXTEND
LIS

EXTHDR
LIS

FILUTL
LIS

GETREQ
LIS

EXTCONTIG
LIS

ERASE
LIS

GETTIM
LIS

FIND
LIS

GETFIB
LIS

INIFC2
LIS

INIFCP
LIS

EXTFCB
LIS

FILESERV
LIS

FILESIZE
LIS

GETPTR
LIS